Volume 1 • Issue 5
March–April 2000

Introduction to Industrial Ethernet, Part 3.

Part 2 was featured in Issue 4, Winter 1999. If you would like a copy, please send your request to info@ccontrol.com

the EXTENSION
A Technical Supplement to control NETWORK

INTRODUCTION TO THE TRANSMISSION CONTROL PROTOCOL

How does TCP and UDP impact control networks?

By George Thomas,
Contemporary Controls

INTRODUCTION

In the last EXTENSION issue (Volume 1 Issue 4) we discussed the impact of the Internet Protocol (IP) on control networks. IP resides at the network layer of the OSI communications model and provides the basic unit of data transfer, which includes addressing, routing, and fragmentation. The transport layer of this same model resides above the network layer and provides station-to-station communication and a common interface to the application layer. This implies reliable communication, which is either accomplished at the transport layer or at the application layer. With control networks this is usually accomplished at the application layer since many control networks were designed before the popularity of TCP/IP took hold. Still there are some control network protocols, such as MODBUS/TCP, which do rely upon the guaranteed delivery mechanism of TCP and there may be more in the future. Actually, at the transport layer of the TCP/IP stack there are two transport protocols, each of which find service in control networks. The User Datagram Protocol (UDP) and the Transmission Control Protocol (TCP) will both be discussed in this article.

USER DATAGRAM PROTOCOL

UDP provides a connectionless and unreliable transport service since it does not issue acknowledgements to the sender upon receipt of data nor does it inform the sender that data was lost. Data integrity can suffer by dropped packets, missequenced packets or by the receipt of duplicate packets. Any of these situations can occur without the knowledge of the sender. It appears that UDP is no better than the IP protocol but there is one big difference. UDP introduces the concept of port numbers, which are used by the application layer that resides above UDP. Port numbers have significance in terms of actions requested by the application itself that require a particular response by the receiving station.

The UDP header is short and simple. Only eight bytes are required in the header. Source and destination ports are each 16-bits long and, therefore, require four bytes. The message length of 16-bits indicates the length of the header and attached data. A 16-bit checksum is used to check the validity of the header and data. The UDP header and attached data, which comes from the application layer, are encapsulated into the IP data field. An IP header, which provides station addressing, precedes the UDP datagram and the complete IP datagram is encapsulated into the frame of the data link layer technology used, such as Ethernet, and sent to the desired station where the complete process is reversed. Notice that the only contribution UDP provided was the assignment of port numbers

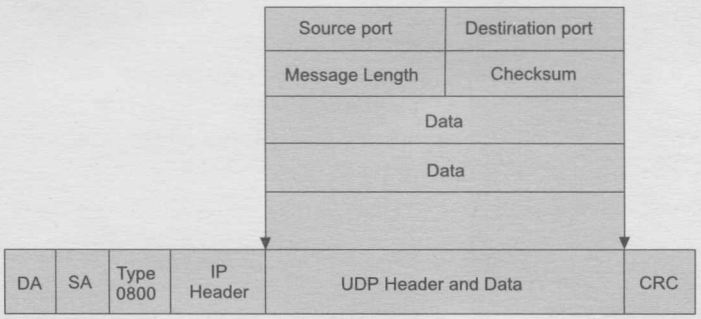

Figure 1. *The UDP header is quite short and, along with its data, it is encapsulated into an IP datagram.*

for use by the application layer. If UDP is to be used, the application layer must worry about acknowledging message receipt, correctly ordering received packets into meaningful messages, discarding duplicate packets and requesting retransmission of faulty packets since UDP does not provide this service. However, if the application layer was originally designed to provide this reliability of service there is no reason to have the transport layer duplicate effort so UDP makes sense. UDP has low overhead and executes quickly making it attractive for control networks.

Port Number Assignments

Both UDP and TCP use port numbers and, if possible, the same assignment is used for both. The 16-bit numbers are classified as either assigned (well-known), registered or dynamic (private). Port number assignments are maintained by the Internet Assigned Numbers Authority (IANA) and the complete list can be found at ftp://ftp.isi.edu/in-notes/iana/ under port numbers. Numbers in the range of zero to 1023 are classified as well-known and are used by common processes. However, individual organizations can register numbers in the range of 1024 to 49151 for proprietary purposes and will not be used by other organizations. Looking at the list, we find some companies in our industry. Opto22 has registered 22000 and 22001 for their SNAP I/O and Opto Control products. The BACnet building automation protocol has registered 47808. DeviceNet and ControlNet intend to use 44818. Interestingly, the MODBUS/TCP specification calls for port 502, which is in the well-known port group. Port numbers 49152 to 65535 are considered as either private or dynamic and can be used by anyone.

PORT NUMBERS

UDP introduces the port number concept. When a station receives a UDP datagram, it serves up the port number to the application layer, which then allocates a buffer area for the attached data. The port number has significance since it identifies a particular application. Since there are many port number possibilities, several different applications can be simultaneously supported on the same station.

Port numbers are 16-bits long and are classified as being assigned, registered or dynamic. Assigned port numbers in the range of zero to 1023 have been defined by the Internet Assigned Numbers Authority (IANA) for various applications that are considered part of the TCP/IP protocol suite. These applications include TELNET, FTP, and other popular Internet applications. These port numbers are termed "well known" and cannot be used by other applications. The remainder of the port assignments is classified as being either registered or dynamic. Registered means that an organization wants to define some level of functionality and has registered a unique port number with the IANA. Other organizations are to respect this assignment and not use either assigned or registered port numbers. Finally, the dynamic port numbers are used in a random manner by a requesting station to signify the source port of an application request.

For example, if station A requests a trivial file transfer protocol (TFTP) service (TFTP happens to use UDP) from station B, it would insert a 69 (a well-known port assignment indicating TFTP services) into its destination port. A dynamic (random but non-conflicting) number will be put in its source port and the request will be sent to station B. Station B would receive the request and recognize it is a TFTP request from station A since port number 69 was used. Station B would then begin executing the process. But it would insert a 69 in its source port, and the dynamic number that was generated by station A in its destination port, and send the response to station A. Station B knows how to handle this particular application since it recognizes both the source and destination port numbers. The use of port numbers along with the IP address creates what is called a socket, which can be represented as <netid, hostid, portid>. As long as there is structure to the assignment of ports, the socket assignment becomes a unique representation of a particular application on the complete IP network.

TRANSMISSION CONTROL PROTOCOL

The second transport layer protocol is TCP which provides for a connection-based reliable message delivery service for processes. This relieves the application layer the responsibility of guaranteed message delivery. Besides reliable connections, TCP provides flow control to ensure stations are not flooded with data.

Data transmitted between stations is sent as a series of packets. These packets are reassembled at the receiving end in the proper order to recreate the data stream that was transmitted. Along the way packets can be corrupted, lost, duplicated or received out of

order. In order to make sense of all this, sequence numbers are applied to each packet transmission. A sequence number is assigned to the first packet of a message. It does not matter what is the initial value of the sequence number only that the second packet has been assigned the initial sequence number plus one. The rule is that successive packets of a data stream have ascending sequence numbers each incremented by one. After all the packets are received, sequence numbers are used to order the packets. Missing sequence numbers indicate the packet was lost. Duplicate packet numbers indicate duplicate packets were received allowing them to be discarded. If all the packets are received in tact then the data stream was received properly and the receiving station could acknowledge receipt to the sender. If not, a request for retransmission of the missing packet can be made. There is no need to resend the entire data stream.

TCP provides a byte-oriented sequencing protocol that is more robust than the packet sequence scheme described above. Instead of sequencing each packet, TCP sequences each byte in the packet. Assigning a sequence number to indicate the first byte in a multi-byte packet does this. The second packet will have a sequence number equal to the first sequence number plus the number of bytes in the first packet. The receiver expects the same. The receiver acknowledges receipt of good packets by transmitting an acknowledgement that includes the value of the next expected sequence number. In this case, it is the sequence number of the last byte in the last received packet plus one. Upon receipt of this acknowledgement, the sender can assume that previous bytes up until the indicated sequence number were received properly. Before a sender can discard the packets being sent it must wait until this acknowledgement has been received because it may be called upon to resend some of the data. Acknowledgements need not require a separate transmission and each packet need not be acknowledged. For efficiency, TCP allows an acknowledgement field to be sent along with data. For example, if station B is responding to station A's request for data, that data can be sent along with an acknowledgement of station A's requesting message. This speeds up processing.

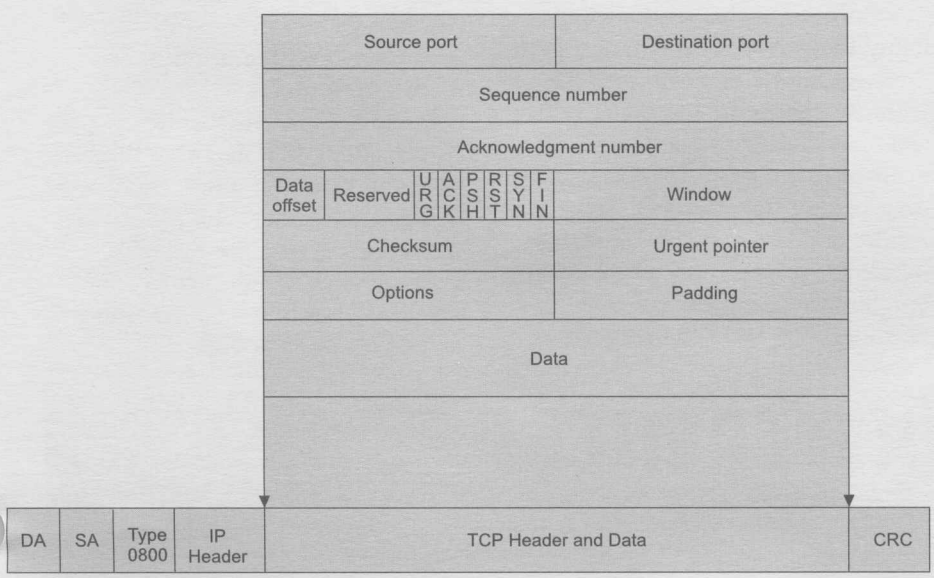

Figure 2. *The TCP header is much more complex than UDP and its length can vary.*

TCP's header is larger than that of UDP but uses the same port assignment scheme as UDP does. Unlike UDP, a 32-bit sequence number and acknowledgement number are included in the header as well as several flag bits. Only a few of the flag bits will be discussed here. Since the TCP header can be of varying length depending upon the content of the options field, a data offset field has been provided in order to determine the actual beginning of the data. The padding field has been provided so that the options field along with the padding field will end on a 32-bit boundary. This is typically done with all the headers within the TCP/IP stack. A window field in the header indicates to the sender how much data the receiver is willing to accept. This feature is used for flow control, which attempts to prevent buffer overflow in the receiver. Finally, data follows the TCP header. The header can be up to 40 bytes in length and the header with its appended data is called a segment. A checksum field ensures the integrity of the header and its associated data.

USING CONNECTIONS

When using the TCP protocol with processes a connection must be first established and maintained in order to provide for the flow of data. By establishing a connection between two stations, the proper buffer area is provided for the impending data. When station A wants to communicate to station B it must first establish a connection which allows for the synchronization of sequence numbers and acknowledgements.

This process is shown in figure 3. One of the flags within the TCP header is the SYN bit, which is used to indicate the initial sequence number when a connection is established. This informs the receiver to synchronize its error checking means to this sequence number. Therefore, station A sends a TCP segment with SYN set and its sequence number, which in this case is 10, to station B. Station B responds by sending an acknowledgement with the value 11 to indicate that is the value of the sequence number it expects to receive next. Station B has its own sequence number that it sends to station A. In this case it is 30 which station A acknowledges by sending out a 31. This establishes two connections. Once the connections are established, data is sent from station A to station B with station A incrementing sequence numbers and station B acknowledging them. This is what is called a full duplex connection since two connections were established; one from A to B and another from B to A. These connections remain established until terminated.

Once the required data transfer has been completed, the two connections should be terminated in order to free up buffer space in the two stations. There is a flag called FIN, which is used to terminate the connection. Station A sends a TCP segment with FIN flag set along with a sequence number. Station B acknowledges the request and once the acknowledgement is received at Station A the connection from A to B is terminated. This does not mean that the connection from B to A is terminated. This must be done in a like fashion in order to terminate the full duplex connection.

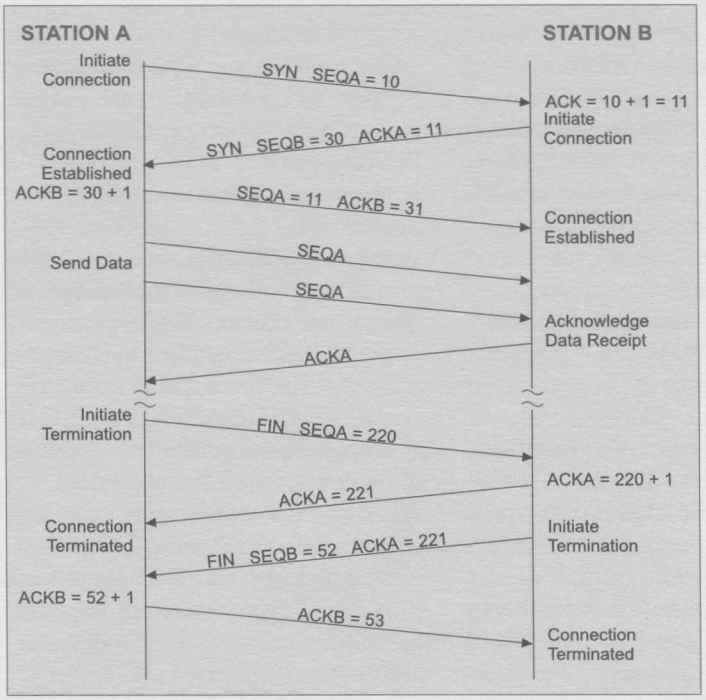

Figure 3. *A full-duplex connection must first be established before data can be transferred using TCP.*

FLOW CONTROL

Flow control is the management of data transfer between two stations. Depending upon the role of the station, be it client or server, or its processing power, a station may not be able to keep up with the network traffic. In order to slow down events the TCP header has a field called window. The receiving station sets a value in the window field informing the sender how many bytes of data it will accept. The window is dynamic and the window can be increased as buffer space in the receiver becomes available. The window can also be zero halting transmission. If the sender still needs to communicate important information while in this condition it can send out a segment with the URG (urgent) flag set along with a sequence number in the urgent pointer field that indicates the first byte of data following the urgent data. The receiver should always allow room for urgent data.

SUMMARY

Although TCP is more reliable than UDP, UDP can be quite effective if the application layer can handle error checking and retransmission. For applications that require secure communication, TCP is the best choice. Both TCP and UDP use port numbers and when used with IP addresses create unique socket definitions across the network. These socket definitions facilitate the processes between the stations on the control network.

REFERENCES

Illustrated TCP/IP, Matthew Naugle, 1998, Wiley Computer Publishing
TCP/IP Clearly Explained, Pete Loshin, 1997, Academic Press